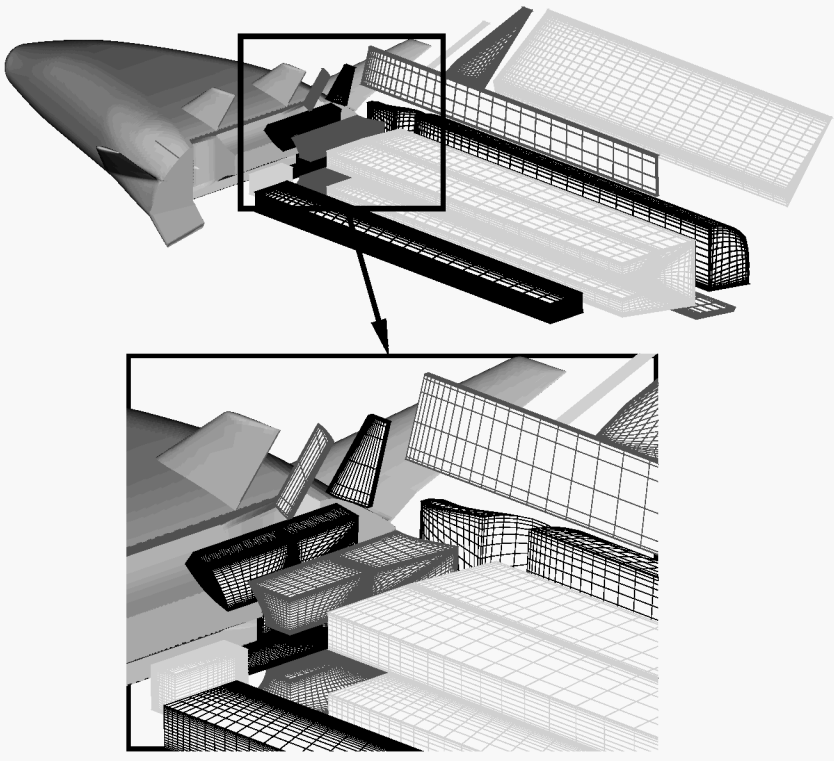
AIAA 98–3012

Grid Generation Techniques Utilizing the Volume Grid Manipulator

Stephen J. Alter
Lockheed Martin Engineering & Sciences
Hampton, Virginia 23681



29th AIAA Fluid Dynamics Conference
June 15–18, 1998/Albuquerque, NM

For permission to copy or republish, contact the American Institute of Aeronautics and Astronautics
1801 Alexander Bell Drive, Suite 500, Reston, VA 20191–4344

Grid Generation Techniques Utilizing the Volume Grid Manipulator

Stephen J. Alter*
Lockheed Martin Engineering & Sciences Hampton, Virginia 23681

This paper presents grid generation techniques available in the Volume Grid Manipulation (VGM) code. The VGM code is designed to manipulate existing line, surface and volume grids to improve the quality of the data. It embodies an easy to read rich language of commands that enables such alterations as topology changes, grid adaption and smoothing. Additionally, the VGM code can be used to construct simplified straight lines, splines, and conic sections which are common curves used in the generation and manipulation of points, lines, surfaces and volumes (i.e., grid data). These simple geometric curves are essential in the construction of domain discretizations for computational fluid dynamic simulations. By comparison to previously established methods of generating these curves interactively, the VGM code provides control of slope continuity and grid point-to-point stretchings as well as quick changes in the controlling parameters. The VGM code offers the capability to couple the generation of these geometries with an extensive manipulation methodology in a scripting language. The scripting language allows parametric studies of a vehicle geometry to be efficiently performed to evaluate favorable trends in the design process. As examples of the powerful capabilities of the VGM code, a wake flow field domain will be appended to an existing X33 VentureStar volume grid; negative volumes resulting from grid expansions to enable flow field capture on a simple geometry, will be corrected; and geometrical changes to a vehicle component of the X33 VentureStar will be shown.

Nomenclature

I,ξ	streamwise computational direction measured from nose to tail of body
J,η	circumferential computational direction measured from top to bottom of body
K,ζ	computational direction normal to body surface
$\Delta S_{i,j}$	distance between points (i,j,k) and $(i,j-1,k)$
X,Y,Z	Cartesian coordinates

Introduction

With the advent of modern super computers and improved algorithms for generating volumetric discretizations about complex aerodynamic configurations, the use of viscous computations in parametric studies to evaluate favorable trends in the design process have become more common.[1–3] By incorporating viscous computations into the design process, the time required to take ideas from paper to a final product has been significantly reduced.[4–6] The development of high fidelity grids for viscous computations using state-of-the art Computer Aided Design (CAD) and grid generation software is still cumbersome and daunting because no single tool has every generation and modification capability. Usually, multiple codes are used to develop a single volume grid, but for competitive reasons, no synergy exists among the various codes. To prevent the loss or corruption of data during transfer of information between the multiple codes, various conversion codes and exchange standards are utilized. This increases the number of codes needed to develop a viscous volume grid.

The side effects of employing multiple codes to generate viscous volume grids are compounded by the inability of most tools to perform three-dimensional manipulations of the points, lines, surfaces, and volumes (.e., grid data). These manipulations may be required to create multiple block interface boundaries and extrapolation or extrusion of existing volume grids to provide adequate flow capture for various Computational Fluid Dynamic (CFD) simulations, such as wakes and shear layers. CAD tools are used to do grid extrusion for the development of wake flow field regions, and extrapolation to expand the outer domain of an existing grid, while grid generation tools are typically used to develop the final grid. Throughout the process of generating the volume grid, manipulations are required to obtain high fidelity surface and volume grids.[4] Development of volume grids used to capture wake domains that append to existing forebody solution-converged volume grids, can be difficult, especially when attempting to maintain outer-domain flowfield capture. Additionally, as the CFD simulation evolves and grid adaption schemes are used to improve the efficient use of grid point, unknown flowfield transients can impede the development of an adapted grid that is free of crossed grid lines and negative cell volumes.

Most of the state-of-the-art domain discretization tools do not offer controls for slope continuity and grid

*Senior Aeronautical Engineer, Senior Member AIAA

Copyright © 1998 Stephen J. Alter. Published by the American Institute of Aeronautics and Astronautics, Inc. with permission.

point-to-point stretchings using vector algebra. Use of existing grid data to generate new data required for the generation high fidelity grids is usually cumbersome as the user has to remember cell sizes along all grid lines in the grid being generated as well as curve information to ensure slope continuity of grid lines at matching boundaries. These problems coupled with the fact that making minor changes to the defining geometries and domain definition curves for the flow domain to be modeled in an interactive environment is slow and tedious, make the grid generation process difficult and time consuming. For example, generating a single face of a computational cube may require the storing of eight cell sizes, one at each end of each surface edge, the point-to-point growth rate for each edge end, and the eight angles at which the connecting grid lines attach to the current surface. The resulting 24 pieces of information are needed to ensure the development of high fidelity grids where point-to-point stretchings, cell sizes and slope continuity control the grid quality. Furthermore, many tools provide a user with the capabilities of generating high fidelity grids efficiently, but the tools usually lack the bare necessities required to provide minor changes in a timely manner.

All of these problems associated with developing high fidelity surface and volume grids for viscous computations can be readily solved through the implementation and use of the Volume Grid Manipulator (VGM) code.[7] The VGM code was initially developed to alter existing line, surface, and volume grids to perform manipulations such as coarse to fine grain grid resolution from grids adapted to capture flow gradients, insertion of vehicle design changes for parametric studies, general removal of negative volumes resulting from grid generation and adaption, and grid smoothing. Through the extensive exercising of the VGM code, methods were discovered to generate grid data. Use of the VGM code can significantly reduce the number of codes used in the grid generation process, minimize the loss of data, and offer easy access to existing grid parameters. It provides the capabilities to do minor changes in defining flowdomain curves without sacrificing quality and resources. When the easy access of grid parameters such as cell sizes, point-to-point stretichings (i.e., cell growth rate) and grid line slope, is coupled with the scripting language, implicit dependencies on modified grids can be created. These dependencies make the regeneration of grid data fast and efficient by limiting the amount of new information required to regenerate any grid. Use of the VGM code for grid generation offers numerous advantages (over methods that employ multiple codes) including:

[1] State-of-the-art grid generation techniques reside in VGM.

[2] The scripting language can be used to significantly reduce the time to re-generate a surface or volume grid.

[3] Extrusion along grid lines is available in a grid generation forum.

[4] Extrapolation can be done using various functions and parametric spline bases.

[5] Smoothing techniques are available to improve slope continuity and quality.

[6] Extraction of grid parameters such as cell sizes, point-to-point stretchings and grid line slope angles to control grid quality, can be done easily and efficiently.

This paper will address various techniques that can be used to perform grid generation and advanced manipulations using the VGM code by comparison to previously established methods. These techniques will be used to circumvent the problems associated with the development of wake volume grids, the expansion of existing grids to ensure outer-domain flowfield capture, and grid adaption without the creation of negative volumes. Examples of each will be shown for either the X33 VentureStar vehicle or a simple geometry.

Example Geometry

Throughout the explanation of the grid and curve generation techniques in this paper and the VGM code, the geometry being used is as shown in Fig. 1. For the coordinate system shown, the I (or ξ)-coordinate increases from the nose to tail, the J (or η)-coordinate increases from the top to the bottom of the vehicle, and the K (or ζ)-coordinate increases from the geometry surface to the outer boundary. The surface geometry represents a sphere, cone, cylinder axis-symmetric geometry. The only complex face that will be addressed in the development of the grids will be the leeside symmetry plane at the minimum J location.

Original Grid Generation Strategies

State-of-the-art grid generation codes used to develop domain discretizations for fluid flow computations utilize and strive towards interactive tools[8–10] for the construction of defining geometry and curves. These interactive tools, implemented in a Graphical User Interface (GUI), are typically used to generate conic sections, splines, and straight lines that initially define the flow domain. Throughout the viscous volume grid construction process it becomes necessary to regenerate these defining curves or modify them to enhance grid fidelity or improve the domain of flow capture. A primary drawback of the GUI approach for grid generation, is the modification of these curves

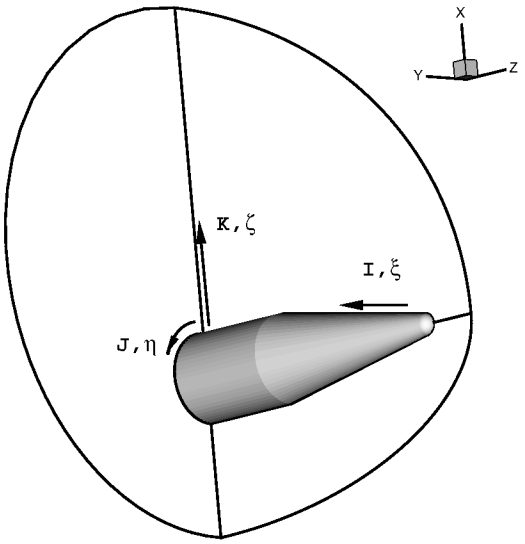


Fig. 1 Grid-point and computational coordinate orientations.

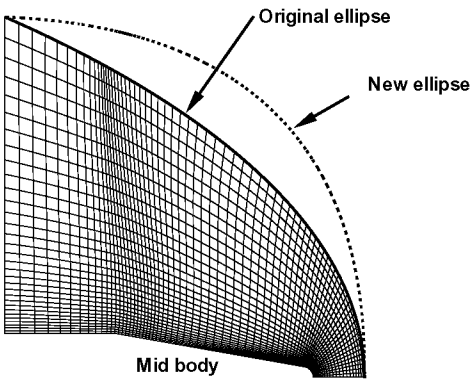


Fig. 2 Outer domain changes to improve flow field capture.

both tedious and time consuming. This drawback arises because curves are usually regenerated as opposed to altered. For example, as shown in Fig. 2 if the ellipse used to define the outer flow domain of the example geometry is not sufficiently expanded in the mid body region, CFD simulations utilizing this grid may not encompass the flow domain at low angles of attack. Generation of this ellipse may only take a few minutes of wall clock time, but subsequent regenerations may take the same amount of time until a suitable domain is obtained. These changes would be necessary if the flow domain was initially not large enough to capture a flow field. Once the changes in the domain definition are made, regeneration of existing volume grid is required to ensure the highest fidelity possible in the volume grid. Thus, by changing a defining curve, the process of updating the volume to reflect the change requires regeneration of the entire domain in which the change occurred.

Another drawback to the use of GUI based grid generation tools is the inability to extract grid parameters from existing data to be used towards the generation of new grids. Extraction of grid parameters such as slope and grid point-to-point stretchings are essential in the development of domain discretizations as they help define subsequent sections of a flow field. Usually these parameters have to be determined manually, which can be cumbersome in 2D and nearly impossible in 3D due to the required computations. For a 2D case, if the example geometry is to have a wake domain attached such that the flow region continues to expand at the current rate and the point spacings are to be nearly monotonic, computations of the grid-line slope at the outer domain and all the cell sizes at the interface need to be done. For each point along the interfacing line, three pieces of information are needed, namely a cell size, a growth rate or point-to-point stretching, and a slope. For example, for 65 points from body to outer domain, 195 pieces of information is required for the extrusion of a 2D case. And for a 3D case, if the geometry is as simple as the example geometry, 37 meridinal planes of data need to be computed, for a total of 7215 bits of information. Remembering all of these values is next to impossible, and storing of them by hand is extremely time consuming. Hence, compromises of grid quality are made to reduce the complexity of the problem, which can adversely affect the accuracy of CFD simulations.[11]

The placement of slope continuous lines at the outer boundary can be done analytically or graphically. As shown with gray lines in Fig. 3, graphical placement of a slope continuous line usually does not result in a correct slope and lines of different slope result from separate attempts. If the fidelity of the grid is to be of the highest possible, computation is required. When the computation of the slope is coupled with the computation of the cell sizes and the stretching derivatives that are required at each streamwise grid line, the time required to extrude the exit domain to create the high fidelity wake domain grid becomes excessive. This construction required two hours of wall clock time for the simple example shape. These computations in three dimensions is prohibitive due to the plethora of data required. Use of another code to determine the controlling parameters is imperative. Further, once the domain is generated, regeneration of the domain may be required if the wake is not sufficiently large enough to capture the flow gradients. This regeneration consumes the same amount of time since most of the controlling parameters have to be recomputed and the domain has to be regenerated.

Regeneration of the volume domain requires the use

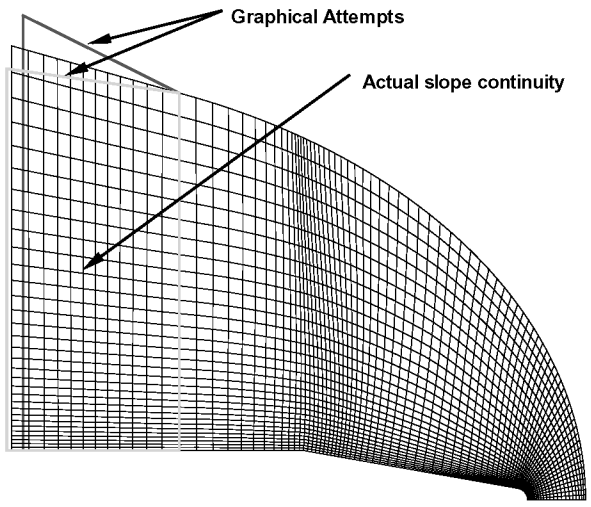


Fig. 3 Wake generation using GUI based tools.

of more tools and resources. As the number of regenerations increases, the time available to perform CFD simulations diminishes and the competitiveness afforded through the use of these tools decays rapidly. Hence, minor modifications to existing flow domains to improve grid quality or the capture of flow fields are usually not performed because they are either too time consuming or not deemed to be worth the enhanced evaluation of design performance data. This reluctance to perform minor changes on flow domains can reduce the capabilities of a design to meet mission goals and can result in loss of revenue from the sales of vehicles that do not perform well by comparison to competitors.

New Grid Generation Strategy

An alternative to using GUI based tools to do grid generation, is the use of a batch or interpreter code such as the Volume Grid Manipulator (VGM). This tool embodies a scripting language that can significantly reduce the time to do minor changes to existing volume grids. It contains 2D and 3D grid generation capabilities as well as smoothing techniques that can be used to easily regenerate the volume containing the minor changes. VGM commands can extract grid parameters such as cell size, spacing gradients, and grid-line slope that can be used in the generation processes.

Three curve types will be addressed in this paper, that can be used in the construction of flowfield domains. Use of the VGM code to generate these curves significantly reduces regeneration time as the scripting language enables rapid turnaround to minor changes in the parameters used to control the curve types. The curves that will be discussed include straight lines that are projected in 2D and 3D, splines, and conic-sections.

There are several commands in VGM, tabulated in Tab. that can be exploited to generate splines, straight lines, and conic sections. Combining these geometrical items with the Trans-Finite Interpolation (TFI) commands can produce surface and volume grid discretizations of new data. Pre-existing grid data on which the grid generation is based, is still required to initiate this process. The redistribution command embodies a method to generate straight lines, while the equating and blending commands offer techniques for spline and conic-section curve generation.

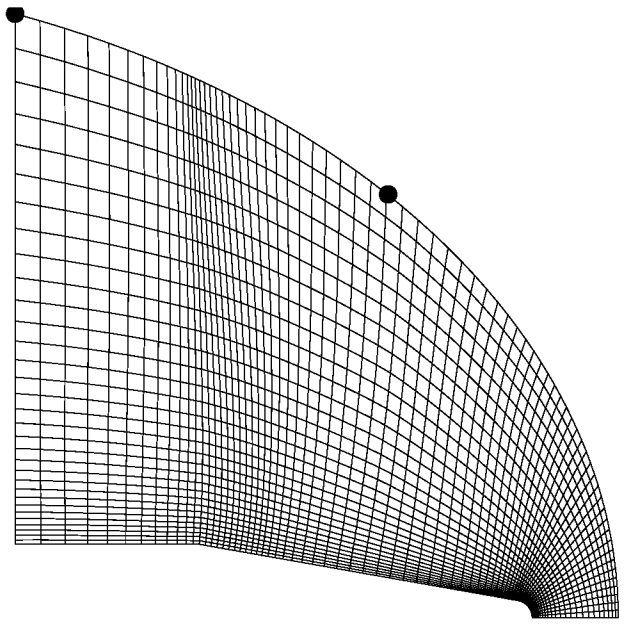
Command	Description
allocate	Create a new block or array variable
blend	Interpolate between existing points in a variable
combine	Regroup volume grids into a single grid system
copydist	Copy a distribution from one grid line to another
for...next	Looping for a repetative complex manipulation
quit	End execution
read	Input data
redist	Redistribute a grid line based on a function
set	Equate variables and data or compute data
smooth	Smooth a grid with algebraic or PDE solvers
tfi	Perform Trans-Finite Interpolation on a region/zone
write	Output data

Table 1 VGM command summary.

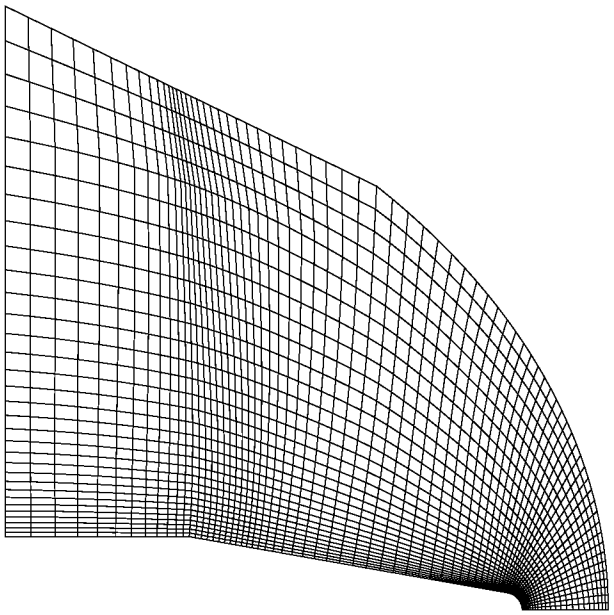
Straight Line Generation Techniques

Straight line grid generation is simple using the redistribution command. As with most VGM commands, this command has several arguments including a grid block specifier. The grid block specifier allows a user to identify the region of a grid to be manipulated using grid point limits and increments. By tailoring these limits and increments to select the beginning and ending points within a grid, the command can interpolate a straight line between the points using a linear basis interpolation function coupled with any number of distribution functions. For example, in the grid illustrated in Fig. 4, a straight grid line is generated at the outer domain region by fixing the endpoints identified by the large dots in Fig. 4(a) and using the redistribution command. Notice that the distribution on the straight line shown in Fig. 4(b) is identical to the original distribution.

Straight line grid generation can also be performed by selecting a root point and projecting a specified distance along a unit vector. The unit vector is obtained using existing grid points as end points for the



a) Original grid.



b) Straight line generation at outer boundary.

Fig. 4 Straight-line generation at an outer boundary edge.

vector. Utilizing the unit vector and a length, the beginning point for an extrusion or extrapolation can be projected to determine the ending point. Then the redistribution command can be used to generate the associated grid line. As an example, if the symmetry plane of a flow domain is to be extruded to generate a domain to capture the wake flow for the example geometry, unit vectors and grid projections can result in the two-dimensional appended grid illustrated in Fig. 5.

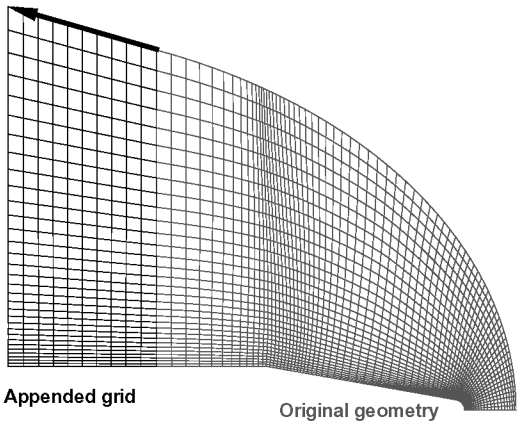


Fig. 5 Projected straight-line generation at an outer boundary edge.

The unit vector for the projection of the outer boundary grid line is shown by the arrow. This projection also determines the downstream distance at which the wall is extended, which is added to the interface grid point at the wall of the sphere-cone. The extruded grid lines have slope continuity at the original exit domain grid line, since the unit vector for the extrusion was determined using the original grid at each point. Slope continuity can also be obtained by using the Hermite Vector Interpolation (HVI) method in VGM to interpolate from the original grid into the wake region.

By comparison to the original techniques of generating this wake with the GUI methods, the VGM method required 10 minutes of wall clock time to generate this domain compared to 120 minutes required by the GUI methods. Subsequent regnerations of the domain for changes in grid dimensionality or domain size can be done more quickly, as these require either simple changes in the controlling parameters or controlling parameteres derived from new geometry. Nearly 80% of the time required to do the generation with VGM was consumed in writing the script; thus subsequent regenerations would only require 1 to 2 minutes. This is a savings of almost two orders of magnitude over the GUI methods, because the GUI methods require manual recomputation of the controlling parameters. Use of the VGM techniques can significantly reduce the time to generate this type of volume grid.

Splined Line Generation Technique

Splined grid line generation is more involved, but straightforward. Instead of generating a grid line in all three coordinate directions at once, each coordinate must be generated separately. The coordinates are separated into single variables where each variable is individually blended to generate new coordinate data. The individual coordinates are then regrouped into the

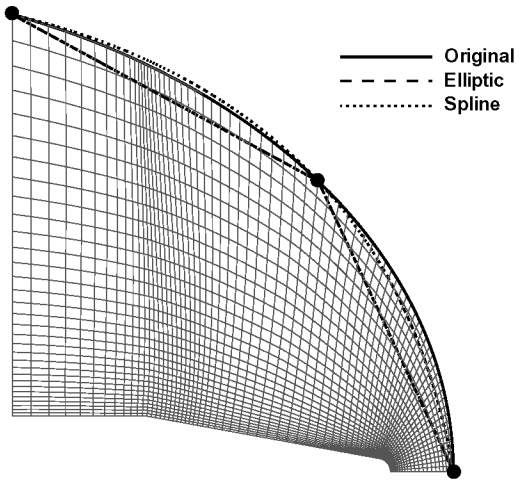


Fig. 6 Spline-line generation at an outer boundary edge.

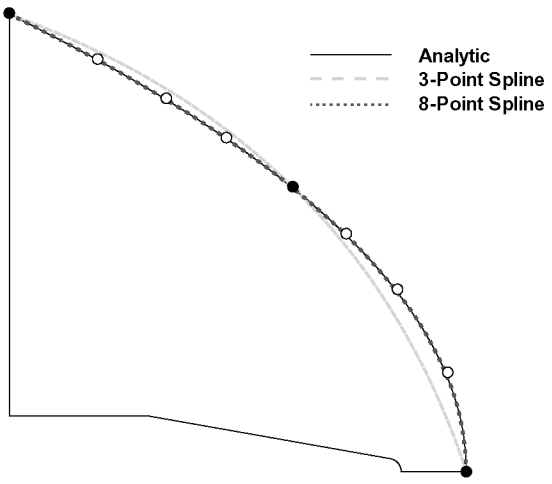


Fig. 7 Elliptic line generation at an outer boundary edge.

three components that comprise a grid. The blending that can be specified in VGM includes straight line interpolation between points and an elliptic and spline basis interpolation function. Splined grid lines can be blended along a curve which contains fixed control points and interpolates between those points with the basis function. For example, Fig. 6 illustrates both the elliptical function and spline generated grid lines to create an outer boundary domain for the example geometry. Notice that the elliptic basis function takes on the attributes of a linear interpolation. This occurs because the blending function is linearly blending between the known data, but the spacing between the points is following an ellipse which has attributes of a hyperbolic tangent function. The fixed points are identified as large solid dots, the elliptical function generated line is illustrated as a dashed line, while the spline line is identified with a dotted line by comparison to their original location identified by the solid line. The generation of the splined line will change based on the interval of the fixed points.

Conic Section Generation Technique

Conic sections are generated similar to the splined lines, but placement of the fixed points has to be done manually. There are capabilities to do the complex arithmetic within the VGM language, but use of the language can be cumbersome for this computation. Aside from the lack of trigonometric and logarithmic functions, the standard multiplication, addition, subtraction and division operations are available. The fixed control points required for conic sections can be generated with these operations, and the scripting language makes the modification of these control points simple and regeneration rapid. Using equations for circles, ellipses, and hyperbolas, the fixed control points, which represent coarse grid definition of these sections, can be quickly defined. The larger the number of fixed points used to define the conic section, the greater the control for the blending. For example, as illustrated in Fig. 7, to place an elliptic segment on the symmetry plane of the example configuration for regenerating an outer boundary grid line, the equation used creates a portion of an ellipse. Typically, this portion produces an orthogonal grid line incidence at the beginning of the configuration and a slope consistent with the flowfield structure at the end of the body.

The only fixed points generated by VGM for this example are identified by the large solid dots in the figure. The open circles indicate fixed points that were added to improve the interpolation. Although this technique requires a bit more work using the limited mathematical operations inside the VGM code, it does prove that the code can generate reasonable grid lines for domain discretizations utilizing the spline basis function.

Application of Generation Techniques

The grid generation techniques available in the VGM code are best explained by examples. This section will address three different applications of VGM grid generation, which highlight the previously discussed techniques. The applications will include:

[1] the development of a wake flow field grid to an existing solution-converged grid for the X33 VentureStar,

[2] the correction of negative volumes generated through the adaption of a grid on the example geometry, and

[3] the generation of a new bodyflap for the X33 VentureStar F-loft configuration.

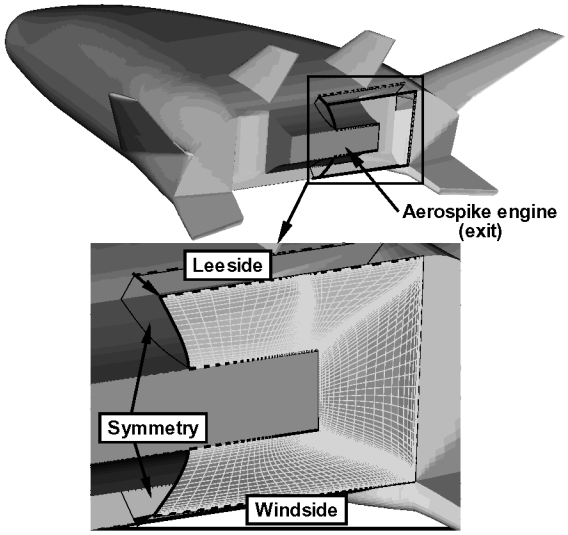


Fig. 8 Interface of the O-grid engine block to the wake.

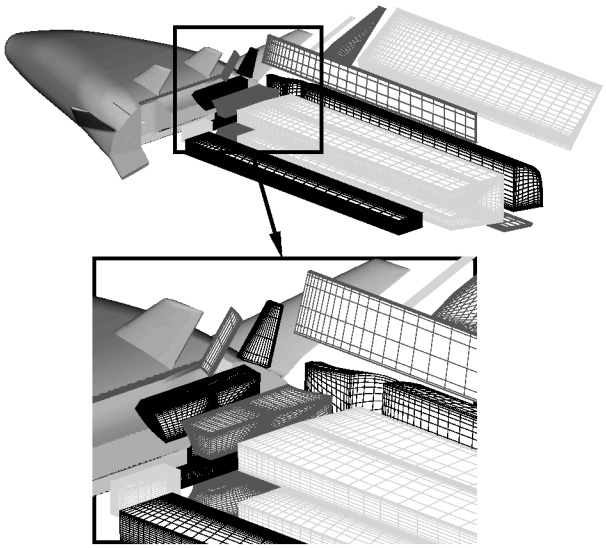


Fig. 9 Expanded view of the X33 wake core blocks.

Development of a Wake Capture Grid

Generation of a wake volume grid for the solution-converged forebody grid of the X33 commences by selecting the topology of the domain to be generated, and the cutoff distance for the wake. The cutoff distance was selected as twice the length of the body, where the separation region produced by the base would be fully captured and an extrapolation boundary condition could be used at the exit of the flow domain. The topology is a multiple block data set comprised of 15 blocks. Each block is generated in a specific order based on the availability of boundary data from newly generated adjacent blocks. Assuming the wall surface grids have been generated, the process of generating the wake blocks begins by constructing the engine block that encompasses the nozzle of the aerospike engine which is darker shaded in Fig. 8. The topology of this block is an O-grid that traverses from the leeside symmetry plane to the windside, as shown in Fig. 8. The connecting edges at the symmetry plane are generated by projecting the base wall grid at the symmetry planes to the end of the engine nozzle using straight lines. The remaining edges are then constructed as straight line connections at the end of the nozzle, with distributions copied from the opposing face edges. The surface that serves as an interface at the end of the engine nozzle does require the use of a 2D grid generator, but is the only surface that requires the additional computing in the wake core blocks. The next blocks that are generated have an H topology and are at the end of the vertical fin, and the end of the wing. These blocks are grouped into the fore-wake region as they complete the computational domain of the forebody.

The next group of blocks are in the mid-wake, which extends from the fore-wake to the end of the bodyflap. These are constructed by generating the block above the bodyflap with straight lines from the base to the end of the flap, and splines on the top of the block. Subsequent blocks of the mid-engine, engine core, mid-fin, and mid-wing are constructed in this order such that the interfaces to each consecutive block serve as known data for consistency in block-to-block matches. The remaining blocks in the wake core, which represent the aft-wake group are generated identically to the mid-wake blocks. Grid point spacing continuity is enforced in all blocks by using the known data of previously generated block interfaces. The final set of wake core blocks is developed iteratively with VGM until monotonicity in point spacing and proper grid clustering for resolving flow gradients about vehicle corners is achieved. The result is the set of blocks shown in Fig. 9.

The final block to be generated is the outer domain that uses the wake core interface, and the exit from the forebody domain, as starting surfaces. The domain is constructed by projecting upstream grid lines on the forebody exit at the symmetry planes and a circular cross-section at the wake exit, as shown in Fig. 10. The dashed lines identify the projected lines used in the formation of the wake, while the dotted lines represent the straight line connections to the wake core. The large dots on the exit cross-section are the control points used in constructing the conic section of a circle for closing off the volume. The remaining undefined surface grids are generated with 2D-TFI and GRIDGEN2D[8] at the exit, and the volume is generated with 3D-TFI. The volume is then improved by running the 3DMAGGS[12] elliptic solver to smooth the grid and obtain orthogonal incidence angles at the interface to the wake core. Representative cross-sectional planes are shown in Fig. 11.

This process initially consumed 18 hours of wall

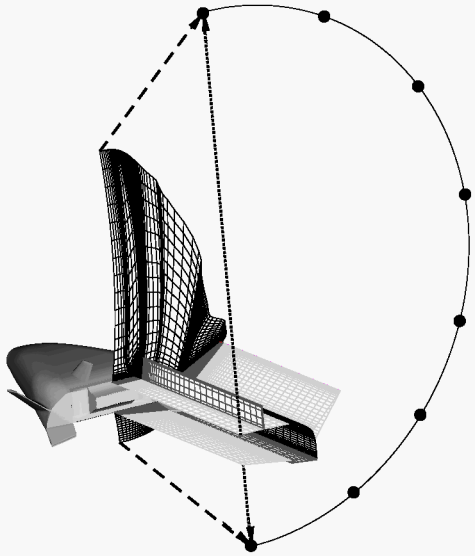
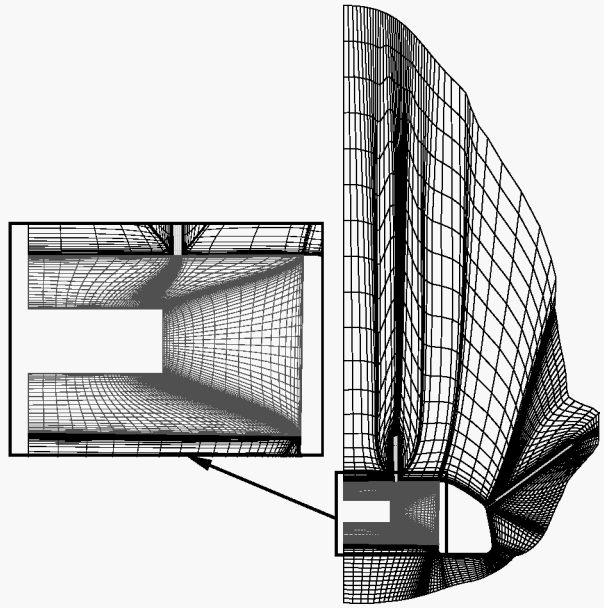


Fig. 10 Projected drive curves used in outer block wake construction.

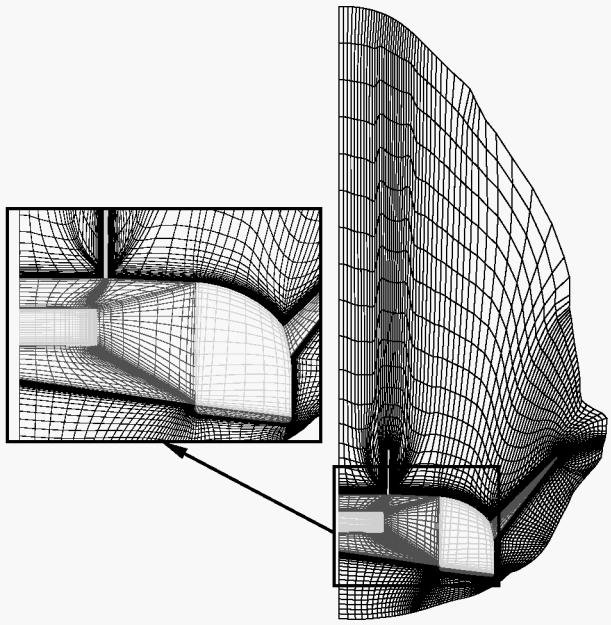
clock time, but a subsequently generated grid to capture the wake took only 2.5 hours because the scripts were already generated, and parameter data from each block was used in the construction. A pseudo parametric grid generation capability is then afforded through the use of the block data as well as the scripts to do the generation. The generation of this domain discretization in a generalized grid generation tool required over 30 hours of wall clock time. Considering the fact that this tool does not have a scripting language, the time to generate each subsequent wake field domain would have required nearly the same amount of time. Therefore, use of the VGM code with its script language and the block data in the construction of the wake flow fields can significantly reduce the time to do parametric studies of wake field phenomenon.

Grid Expansion with Negative Volume Correction

Another application of the grid generation methods of the VGM code is the expansion of a volume grid to enable flow field capture and to correct the negative volumes that can result. The volume grid for the example configuration, shown in Fig. 12 will be expanded to illustrate this grid generation technique, as well as the results of correcting negative volumes produced by the exansion. The expansion is accomplished though straight line extrapolation at the outer domain, on the grid shown in Fig. . As shown in the figure, negative volumes are generated after the expansion where the grid lines are convergent at the outer boundary. These volumes are corrected by first generating a spline that smoothes over the section then using 2D-TFI on the faces and 3D-TFI on the volume to re-generate the volume grid. The results of these corrections are shown in Fig. 14.



a) Fore-wake plane at $I = I_{max}$



b) Mid-wake plane at $I = I_{max}$

Fig. 11 Representative planes of the X33 wake.

Generation of a Vehicle Component

The final application of VGM grid generation techniques is the generation of geometry for the X33 bodyflap. The original bodyflap was deemed to be ineffective based on the size. To test various theories on improving the effectiveness, the size of the bodyflap had to change. Two different shapes were attempted; extension of the control surface, and a coupled extension and widening of the control surface. Each of these changes, while small in the context of the

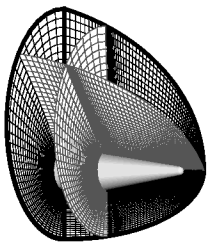


Fig. 12 Original simple geometry volume grid.

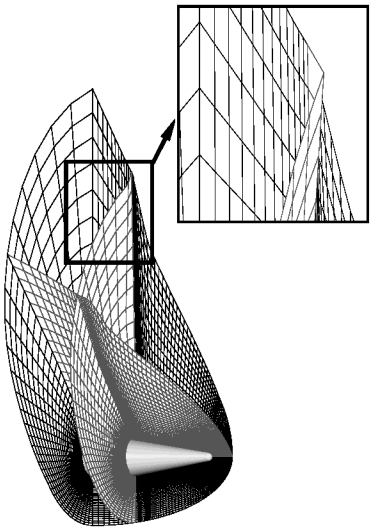


Fig. 13 Expanded grid via straight line extrapolation.

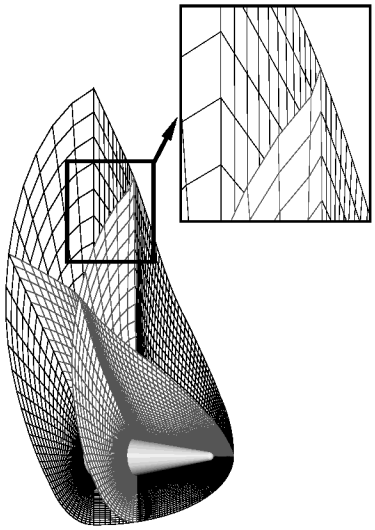


Fig. 14 Corrected negative volumes.

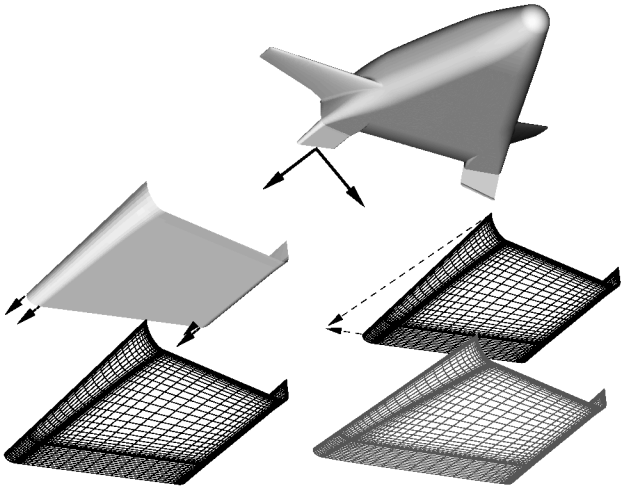


Fig. 15 Extension and expansion of a bodyflap for the X33.

overall X33 geometry, are significant when it comes to building grid models. The VGM code scripting capability avoids regeneration of the complete grid, as would have to be accomplished by most current grid generation methods. This capability represents a considerable savings in time and effort.

Each of these geometric changes are accomplished with the straight line projection algorithm. As shown in Fig. 15 the extension is accomplished by projecting the endpoints and the corners at the ends of the flat section aft a prescribed distance. The cross-sectional grid line at the end of the bodyflap is extended by blendings of the projections' distances between the endpoints. The intervals of grid points that are blended are at the edges and the flat region of the bodyflap. The projection distances of these points are blended by interpolating the distances at the drive curves, identified in dashed lines, for the unknown data. By blending the differences in each coordinate separately, two problems are thwarted: improper projection of all grid lines that may produce negative volumes from grid-line crossing, and the non-linear effects of projecting all the points to create a suitable end to the bodyflap. The blending also reduces the number of computations required to obtain the end of the bodyflap.

The widening is done by projecting the outboard section a specified distance and blending the movements from the aft to the stationary forward section. A linear blend in the arclength domain along the flap leading edge is used to retain a straight line. This method is very effective in that the blendings of the projections are not limited to a cross-section, but can be applied to entire sections of a grid. This is one of the strengths of the VGM code, the ability to manipulate and generate grids in all computational directions.

Conclusions

Three methods of generating grid lines using the VGM code are presented. When these methods are combined with the generation capabilities of TFI and the smoothing capabilities of parametric re-mapping as well as HVI, the VGM code can be used to generate new volume grids based on existing grid data. The scripting language of the VGM code, coupled with the capacity to extract parametric data from the existing grid data, offers a powerful tool to perform parametric studies. These characteristics of VGM provide quick and efficient generation high fidelity surface and volume grids for CFD simulations. Applications of wake field generation, computational domain expansions, and geometry generation demonstrate the versatility and capabilities of the VGM code. Most importantly, implementation of the VGM code in the grid generation process can significantly reduce the time to perform grid generation and topology alterations.

Acknowledgment

This work was performed under NASA Langley Research Center contract NAS1-96014. The author wishes to express his appreciation to Dr. David Schuster and Dr. Ramadas Prabhu for their editorial comments, William Kleb for his contributions to the typesetting of this document and K. James Weilmuenster for permitting this work to be done.

References

[1]Alter, S. J. and Cheatwood, F. M., "Elliptic Volume Grid Generation for Viscous Computations in Parametric Design Studies," AIAA Paper 96–1999, June 1996.

[2]Korpus, R., "Improved Methods for the Orthogonal Control of Highly Clustered Elliptically Generated Grids," *Software Systems for Surface Modeling and Grid Generation*, edited by R. E. Smith, Vol. CP–3143, NASA, 1992.

[3]Vatsa, V. N., Sanetrik, M. D., and Parlette, E. B., "Block-Structured Grids for Complex Aerodynamic Configurations: Current Status," *Surface Modeling, Grid Generation, and Related Issues in Computational Fluid Dynamic (CFD) Solutions*, edited by Y. K. Choo, Vol. CP–3143, NASA, 1995, pp. 163–177.

[4]Gnoffo, P. A., Weilmuenster, K. J., Hamilton II, H. H., Olynick, D. R., and Venkatapathy, E., "Computational Aerothermodynamic Design Issues for Hypersonic Vehicles," , No. 97–2473, June 1997.

[5]Yamamoto, Y., Wada, Y., and Yoshioka, M., "Hypersonic CFD Analysis for the Aerothermodynamic Design of HOPE," , No. 95–1770, June 1995.

[6]Yamamoto, Y. and Yoshioka, M., "CFD and FEM Coupling Analysis of OREX Aerothermodynamic Flight Data," , No. 95–2087, June 1995.

[7]Alter, S. J., "The Volume Grid Manipulator (VGM): A Grid Reusability Tool," NASA CR–4772, April 1997.

[8]J. P. Steinbrenner, Chawner, J. R., and Fouts, C. L., "The GRIDGEN 3D Multiple Block Grid Generation System," Wright Research and Development Center Report WRDC–TR–90–3022, October 1989.

[9]Thompson, J. F., "A Composite Grid Generation Code for General 3D Regions–the EAGLE Code," *AIAA Journal*, Vol. 26, No. 3, March 1988, pp. 1–10.

[10]Chan, W. M., Chiu, I., and Buning, P. G., "User's Manual for the HYPGEN Hyperbolic Grid Generator and the HGUI Graphical User Interface," NASA TM–108791, October 1993.

[11]Olynick, D. R., "Importance of 3–D Grid Resolution and Structure for Calculating Reentry Heating Environments," AIAA Paper 96–1857, June 1996.

[12]Alter, S. J. and Weilmuenster, K. J., "The Three–Dimensional Multi–block Advanced Grid Generation System (3DMAGGS)," NASA TM–108985, April 1993.